

[illegible]

```
CCCCCCCC 000000 MM MM RRRRRRRR 5555555555 000000 AAAAAA SSSSSSSS CCCCCCCC
CCCCCCCC 000000 MM MM RRRRRRRR 5555555555 000000 AAAAAA SSSSSSSS CCCCCCCC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SS CC
CC CC 00 00 MM MM RRRRRRRR 555555 55 00 0000 AA AA SS CC
CC CC 00 00 MM MM RRRRRRRR 555555 55 00 0000 AA AA SS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SSSSSS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SSSSSS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SSSSSS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SSSSSS CC
CC CC 00 00 MM MM RRRRRRRR 55 55 00 00 AA AA SSSSSS CC
CCCCCCCC 000000 MM MM RRRRRRRR 555555 55 000000 AA AA SSSSSSSS CCCCCCCC
CCCCCCCC 000000 MM MM RRRRRRRR 555555 55 000000 AA AA SSSSSSSS CCCCCCCC
```

```
LL LL SSSSSSSS
LL LL SSSSSSSS
LL LL SS
LL LL SS
LL LL SS
LL LL SS
LL LL SSSSSS
LL LL SSSSSS
LL LL SS
LL LL SS
LL LL SS
LL LL SS
LLLLLLLLLL SSSSSSSS
LLLLLLLLLL SSSSSSSS
```


(2) 50
(3) 64
(4) 149

HISTORY ; Detailed Current Edit History
DECLARATIONS
R50ASC - RADIX 50 TO ASCII CONVERSION ROUTINE


```
0000 1 .TITLE COMSR50ASC ; FORTRAN COMPATIBILITY - RADIX 50 TO ASCII CONVERSI
0000 2 .IDENT /1-003/ ; File: COMR50ASC.MAR Edit: SBL1003
0000 3
0000 4 :
0000 5 :*****
0000 6 :*
0000 7 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 :* ALL RIGHTS RESERVED.
0000 10 :*
0000 11 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 :* TRANSFERRED.
0000 17 :*
0000 18 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 :* CORPORATION.
0000 21 :*
0000 22 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 :*
0000 25 :*
0000 26 :*****
0000 27 :
0000 28 :
0000 29 : FACILITY: FORTRAN COMPATIBILITY LIBRARY
0000 30 :++
0000 31 : ABSTRACT:
0000 32 :
0000 33 : R50ASC is a FORTRAN callable procedure to convert radix-50 strings
0000 34 : into ASCII Hollerith strings.
0000 35 :
0000 36 :--
0000 37 :
0000 38 : VERSION: 0
0000 39 :
0000 40 : HISTORY:
0000 41 :
0000 42 : AUTHOR:
0000 43 : Peter Yuo, 9-Sep-77: Version 0
0000 44 :
0000 45 : MODIFIED BY:
0000 46 :
0000 47 :
0000 48 :
```


COMSR50ASC
1-003

6 16
; FORTRAN COMPATIBILITY - RADIX 50 TO AS 15-SEP-1984 23:48:11 VAX/VMS Macro V04-00 Page 2
HISTORY ; Detailed Current Edit History 6-SEP-1984 10:53:13 [FORRTL.SRC]COMR50ASC.MAR;1 (2)

```
0000 50      .SBTTL  HISTORY      ; Detailed Current Edit History
0000 51
0000 52
0000 53 ; Edit History for Version 01 of R50ASC
0000 54 ;
0000 55
0000 56 : 0-03  Correct constant values for EDIV instructions
0000 57 : 00-06 - Change file name to COMR50ASC.MAR, PSECT to F4PCOMPAT. TNH 5-Jan-78
0000 58 : 00-07 - Bug fix: don't use no_output_char for input count! JMT 13-Feb-78
0000 59 : 1-001 - Update version number and copyright notice. JBS 16-NOV-78
0000 60 : 1-002 - Add "" to PSECT directive. JBS 21-DEC-78
0000 61 : 1-003 - Fix bug where largest possible RAD50 value was considered
0000 62 :      invalid. SBL 6-May-80
```



```
0000 64 .SBTTL DECLARATIONS
0000 65
0000 66 :
0000 67 : INCLUDE FILES:
0000 68 :
0000 69 :
0000 70 :
0000 71 : EXTERNAL SYMBOLS:
0000 72 :
0000 73 :
0000 74 :
0000 75 : MACROS:
0000 76 :
0000 77 :
0000 78 :
0000 79 : PSECT DECLARATIONS:
0000 80 :
00000000 81 : .PSECT _F4PCOMPAT$CODE PIC,USR,CON,REL,LCL,SHR,EXE,RD,NOWRT
0000 82 :
0000 83 :
0000 84 : EQUATED SYMBOLS:
0000 85 :
0000 86 :
00000004 0000 87 no_output_char = 4 ; no_output_char.rw.r
00000008 0000 88 radix50_array = 8 ; radix50_array.rbu.ra
0000000C 0000 89 ascii_array = 12 ; ascii_array.wbu.ra
00000640 0000 90 L_50_POWER_2 = ^X0a16 + ^X640 ; 50(octal)**2
0000 91 :
0000 92 :
0000 93 :
0000 94 : OWN STORAGE:
0000 95 :
0000 96 :
0000 97 ASCII_TAB:
20 0000 98 .BYTE ^040
41 0001 99 .BYTE ^0101
42 0002 100 .BYTE ^0102
43 0003 101 .BYTE ^0103
44 0004 102 .BYTE ^0104
45 0005 103 .BYTE ^0105
46 0006 104 .BYTE ^0106
47 0007 105 .BYTE ^0107
48 0008 106 .BYTE ^0110
49 0009 107 .BYTE ^0111
4A 000A 108 .BYTE ^0112
4B 000B 109 .BYTE ^0113
4C 000C 110 .BYTE ^0114
4D 000D 111 .BYTE ^0115
4E 000E 112 .BYTE ^0116
4F 000F 113 .BYTE ^0117
50 0010 114 .BYTE ^0120
51 0011 115 .BYTE ^0121
52 0012 116 .BYTE ^0122
53 0013 117 .BYTE ^0123
54 0014 118 .BYTE ^0124
55 0015 119 .BYTE ^0125
56 0016 120 .BYTE ^0126
```

```
: 0 - space
: 1 - 'A'
: 2 - 'B'
: 3 - 'C'
: 4 - 'D'
: 5 - 'E'
: 6 - 'F'
: 7 - 'G'
: 10 - 'H'
: 11 - 'I'
: 12 - 'J'
: 13 - 'K'
: 14 - 'L'
: 15 - 'M'
: 16 - 'N'
: 17 - 'O'
: 20 - 'P'
: 21 - 'Q'
: 22 - 'R'
: 23 - 'S'
: 24 - 'T'
: 25 - 'U'
: 26 - 'V'
```


57	0017	121	.BYTE	^0127	:	27	-	'w'
58	0018	122	.BYTE	^0130	:	30	-	'x'
59	0019	123	.BYTE	^0131	:	31	-	'y'
5A	001A	124	.BYTE	^0132	:	32	-	'z'
24	001B	125	.BYTE	^044	:	33	-	'\$'
2E	001C	126	.BYTE	^056	:	34	-	'.'
3F	001D	127	.BYTE	^077	:	35	-	unused
30	001E	128	.BYTE	^060	:	36	-	'0'
31	001F	129	.BYTE	^061	:	37	-	'1'
32	0020	130	.BYTE	^062	:	40	-	'2'
33	0021	131	.BYTE	^063	:	41	-	'3'
34	0022	132	.BYTE	^064	:	42	-	'4'
35	0023	133	.BYTE	^065	:	43	-	'5'
36	0024	134	.BYTE	^066	:	44	-	'6'
37	0025	135	.BYTE	^067	:	45	-	'7'
38	0026	136	.BYTE	^070	:	46	-	'8'
39	0027	137	.BYTE	^071	:	47	-	'9'
	0028	138						
	0028	139						

0028 140 : REGISTER USAGE:

0028	141	:	
0028	142	:	R0:R1 - current word for processing (use quadword in order to use EDIV)
0028	143	:	R2 - holds remainder for EDIV
0028	144	:	R3 - CHARS_REM, remaining number of output chars to produce
0028	145	:	R4 - NEXT_INPUT_POSITION, next radix-50 input position
0028	146	:	R5 - NEXT_OUTPUT_POSITION, next ASCII output position
0028	147	:	


```
0028 149      .SBTTL R50ASC - RADIX 50 TO ASCII CONVERSION ROUTINE
0028 150
0028 151      :++
0028 152      : FUNCTIONAL DESCRIPTION:
0028 153      :
0028 154      :   Algorithmic steps:
0028 155      :   1) Initialization
0028 156      :       CHARS_REM = no_output_char.rw.r
0028 157      :       NEXT_INPUT_POSITION = radix50_array.rbu.ra
0028 158      :       NEXT_OUTPUT_POSITION = ascii_array.wbu.ra
0028 159      :   2) If CHARS_REM =< 0 then DONE
0028 160      :   3) Get current word for processing
0028 161      :   4) If overflow then (output '???'', go to step 6)
0028 162      :   5) Do conversion
0028 163      :       [Note] n is a word value, so longword division will not produce
0028 164      :       negative results
0028 165      :       a. (Let n = a2*50**2 + a1 * 50 + a0)
0028 166      :           Divide n by 50**2. (n = q1*50**2 + r1)
0028 167      :           ASCII_TAB[q1] is 1st ASCII char to output
0028 168      :       b. Divide r1 by 50. (r1 = q2*50 + r2)
0028 169      :           ASCII_TAB[q2] is 2nd ASCII char to output
0028 170      :           ASCII_TAB[r2] is 3rd ASCII char to output
0028 171      :   6) CHARS_REM = CHARS_REM - 3
0028 172      :   7) Go back to step 2
0028 173      :
0028 174      : CALLING SEQUENCE:
0028 175      :
0028 176      :   CALL R50ASC (no_output_char.rw.r, radix50_array.rbu.ra,
0028 177      :               ascii_array.rbu.ra)
0028 178      :
0028 179      : INPUT PARAMETERS:
0028 180      :
0028 181      :   no_output_char.rw.r      ; the max number of output chars to produce
0028 182      :   radix50_array.rbu.ra    ; address of radix-50 input location
0028 183      :
0028 184      : IMPLICIT INPUTS:
0028 185      :   NONE
0028 186      :
0028 187      : OUTPUT PARAMETERS:
0028 188      :
0028 189      :   ascii_array.wbu.ra      ; address of ascii output location
0028 190      :
0028 191      : IMPLICIT OUTPUTS:
0028 192      :   NONE
0028 193      :
0028 194      : COMPLETION CODES:
0028 195      :   NONE
0028 196      :
0028 197      : SIDE EFFECTS:
0028 198      :   NONE
0028 199      :
0028 200      :--
0028 201      :
0028 202      :
0028 203      :
0028 204      :
403C 0028 205      .ENTRY R50ASC, ^M<IV, R2, R3, R4, R5>
```



```
002A 206 ; standard call-by-reference entry
002A 207 ; enable integer overflow
002A 208
002A 209 : Initialization
002A 210 :
002A 211
53 04 BC 3C 002A 212 MOVZWL @no_output_char(AP), R3 ; R3 = CHARS_REM
54 08 AC D0 002E 213 MOVL radix50_array(AP), R4 ; R4 = NEXT_INPUT_POSITION
55 0C AC D0 0032 214 MOVL ascii_array(AP), R5 ; R5 = NEXT_OUTPUT_POSITION
51 D4 0036 215 CLRL R1 ; R1 always zero
0038 216
0038 217 : If CHARS_REM <= 0, then DONE
0038 218 :
0038 219 :
0038 220
0038 221 AGAIN:
53 D5 0038 222 TSTL R3
3E 15 003A 223 BLEQ DONE
003C 224
003C 225 : Get current word for processing
003C 226 :
003C 227 :
003C 228
50 84 3C 003C 229 MOVZWL (R4)+, R0 ; R0/R1 = current word for processing
003F 230 ; use quadword for EDIV
003F 231
003F 232 : If overflow then output '???'
003F 233 :
003F 234 :
003F 235
0000F9FF 8F 50 D1 003F 236 CMPL R0, #^0174777 ; 174777(octal) is largest radix 50
0046 237 ; value in a word
50 0000B9E5 07 15 0046 238 BLEQ GOOD ; branch if value is good
8F D0 0048 239 MOVL #^0134745, R0 ; RAD50 for '???'
004F 240
004F 241 : Do actual conversion
004F 242 :
004F 243 :
004F 244
53 D7 004F 245 GOOD: DECL R3 ; dec byte count
27 19 0051 246 BLSS DONE ; and quit if it was 0
50 52 50 00000640 8F 7B 0053 247 EDIV #L_50_POWER_2, R0, R2, R0 ; divide current word (n) in R0/R1 by
005C 248 ; 50(octal)**2,
005C 249 ; n = q1 * 50(octal)**2 + r1
005C 250 ; R2 = q1, R0 = r1
005C 251
85 A0 AF42 90 005C 252 MOVB ASCII_TAB[R2], (R5)+ ; output corresponding ascii char
53 D7 0061 253 DECL R3 ; dec byte count again
15 19 0063 254 BLSS DONE ; and branch if it was 0
50 52 50 28 7B 0065 255 EDIV #^050, R0, R2, R0 ; q1 = q2 * 50(octal) + r2
006A 256 ; R0/R1 = q1, R2 = q2, R0 = r2
85 92 AF42 90 006A 257 MOVB ASCII_TAB[R2], (R5)+ ; output 2nd char
53 D7 006F 258 DECL R3 ; last byte for this RAD50 word
07 19 0071 259 BLSS DONE ; done if it was 0
85 89 AF40 90 0073 260 MOVB ASCII_TAB[R0], (R5)+ ; output 3rd char
BE 11 0078 261 BRB AGAIN ; get next input RAD50 word
007A 262
```


COMSR50ASC
1-003

16
: FORTRAN COMPATIBILITY - RADIX 50 TO AS 15-SEP-1984 23:48:11 VAX/VMS Macro V04-00
R50ASC - RADIX 50 TO ASCII CONVERSION RO 6-SEP-1984 10:53:13 [FORRTL.SRC]COMR50ASC.MAR;1 Page 7
(4)

```
007A 263 :  
007A 264 : DONE  
007A 265 :  
04 007A 266 DONE: RET  
007B 267  
007B 268 .END
```


AGAIN	00000038	R	01
ASCII_ARRAY	= 0000000C		
ASCII_TAB	00000000	R	01
DONE	0000007A	R	01
GOOD	0000004F	R	01
L_50_POWER_2	= 00000640		
NO_OUTPUT_CHAR	= 00000004		
R50ASC	00000028	RG	01
RADIX50_ARRAY	= 00000008		

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes														
ABS	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
_F4PCOMPAT\$CODE	0000007B (123.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	BYTE				

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.10	00:00:01.11
Command processing	105	00:00:00.51	00:00:05.65
Pass 1	70	00:00:00.71	00:00:02.52
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	57	00:00:00.48	00:00:02.52
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	268	00:00:01.85	00:00:11.86

The working set limit was 900 pages.
 3027 bytes (6 pages) of virtual memory were used to buffer the intermediate code.
 There were 10 pages of symbol table space allocated to hold 9 non-local and 0 local symbols.
 268 source lines were read in Pass 1, producing 10 object records in Pass 2.
 0 pages of virtual memory were used to define 0 macros.

! Macro library statistics !

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[SYSLIB]\$TARLET.MLB;2	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:COMR50ASC/OBJ=OBJ\$:COMR50ASC MSRC\$:COMR50ASC/UPDATE=(ENH\$:COMR50ASC)

0178

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY